Collaborative Information Portal: MER and Beyond

Joan D. Walton
NASA/Ames Research Center
joan.d.walton@nasa.gov

Ronald L. Mak
USRA/RIACS
rmak@mail.arc.nasa.gov

Leslie E. Keely
NASA/Ames Research Center
leslie@email.arc.nasa.gov

Abstract

We describe the architecture and interface of the Collaborative Information Portal (CIP), a system that integrates operational and scientific information for managing the 2003 Mars Exploration Rovers (MER) mission. CIP displays schedules, notifies users of events and the arrival of scientific data products, displays the products, and facilitates team collaboration—all within the context of user-personalized access and interfaces. CIP is an Internet-enabled Java desktop application that connects via secure web services to a middleware based on Enterprise JavaBeans (EJB) and a back end containing databases and meta-databases. The system is fully integrated with the current JPL/MER secure flight operations environment and has great potential for use on future planetary missions. Future missions that employ data repositories similar to those used on MER will be able to use CIP with minimal or no modifications. For non-MER type repositories, data integration modules can be written and easily plugged into the architecture without requiring changes to the middleware and client application components. Finally, the CIP architecture provides a basis upon which to build in additional features and functionality beyond the capabilities provided to MER.

1. Introduction

In June 2003, two state-of-the-art Mars rovers launched from Cape Canaveral and started their seven-month journey to the red planet. Once they touch down on the Martian surface in January 2004, they will proceed to deploy their suite of sophisticated scientific instruments to collect geological and meteorological data on the surrounding environment. The goal of the Mars Exploration Rovers (MER) mission is to "follow the water" in an effort to determine whether conditions on Mars may have once been favorable for life [1].

Back on Earth, a team of about 250 scientists and engineers will work around the clock to analyze the collected data, determine a strategy and activities for the next day and then carefully compose the command sequences that will instruct the rovers in how to perform their tasks. The scientists and engineers must work closely together to balance the science objectives with the engineering constraints so that the mission achieves its goals safely and quickly [2]. To accomplish this coordinated effort, they must adhere to a tightly orchestrated schedule of meetings and processes. To keep on time, it is critical that all team members are aware of what is happening, know how much time they have to complete their tasks, and can easily access the information they need to do their jobs.

The Collaborative Information Portal (CIP) addresses the MER mission need for situational awareness and data access by providing a centralized, one-stop delivery platform integrating science and engineering data from several distributed heterogeneous data sources.

2. MER Mission Needs

The specific needs of the MER mission can be broken down into the following categories:

2.1. Time Management

Knowing the current time would seem to be a simple problem solved by installation of a wall clock. However, time will not be so easy to handle on the MER mission, as the mission will operate on Mars time. Since a Martian day, or "sol," is 24.66 hours long, the effect of working on Mars time is that, for mission personnel, each work day will start 40 minutes later than on the preceding day. Over the course of the 90-day mission, the daily schedule will gradually shift through all possible times of the Earth day, and the mission personnel must follow along. Thus, knowing what time to start and end work each day becomes a non-trivial task.

2.2. Personnel Management

To execute the schedule, individual staff members need to know when they are working, in what role, and with whom. Once the staff has made it to work on time, they then need to determine what is happening and where they need to be. The large MER team is divided across three floors of a high rise, each subdivided into rooms designated for various purposes. Added to the time zone challenge, communicating the current schedule to all the team members is complicated by the distribution of their physical locations.

2.3. Data Management

The staff on each MER-mission sol will work on three overlapping shifts spanning the total Martian day. Following the planned process, each shift must hand over

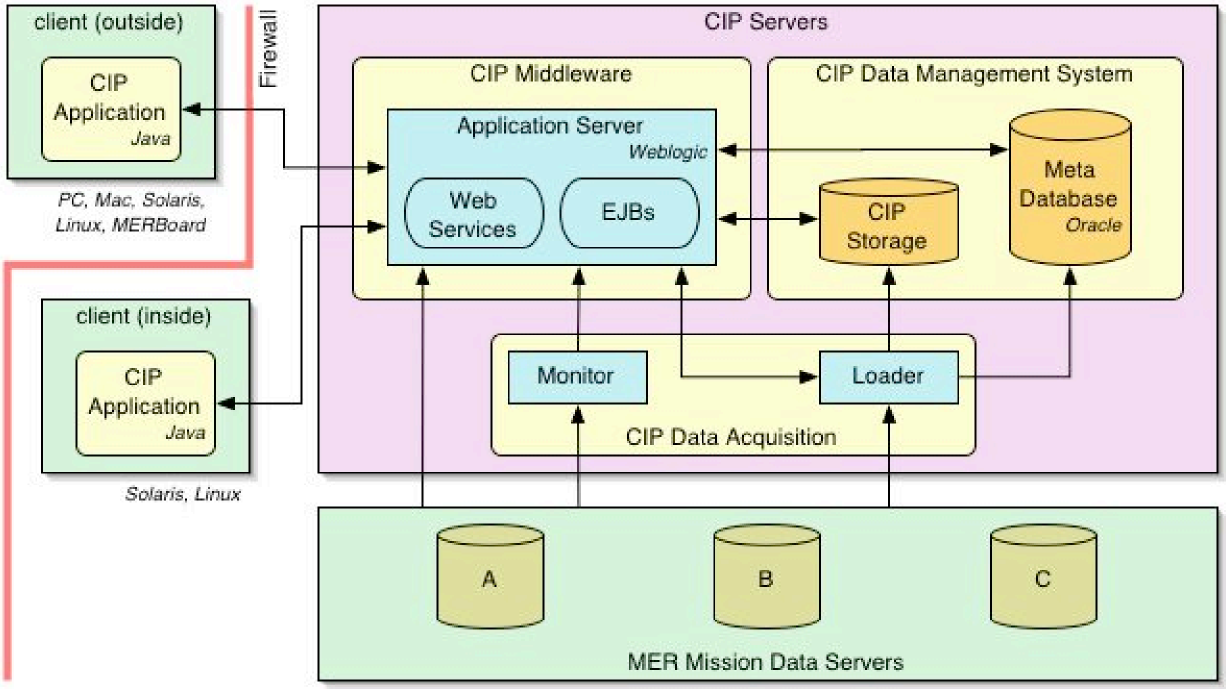

Figure 1. CIP architecture

to the following one as the day progresses from downlink analysis to activity planning to rover navigation and instrument deployment. When a scientist or engineer comes on shift, he or she needs to know what was planned and what actually happened: What was the final plan at the end of the previous sol? Did the rover successfully execute the plan? What just happened on the previous shift? Next he or she needs to locate the data as soon as it becomes available. This activity is complicated by the immense data repository, security restrictions on repository access, rigid structuring of the repository, heterogeneous data products and reports generated by the various specialized subsystems, and the lack of a unified data product notification system.

3. Information Technology Challenges

A solution that addresses the MER mission information-management needs is a unified system that coordinates, facilitates and manages the information flow and is accessible by all the mission staff. The key information technology challenges involved in developing such a system include:

- Integrating heterogeneous data sources.
- Managing large amounts of data.
- Supporting the use of unstructured data.
- Controlling access to data in a distributed and possibly federated environment according to the rights and privileges of particular users.
- Facilitating collaboration.
- Providing tools for browsing and analyzing a range of data.
- Presenting quality interfaces for the above tasks.
- Doing all this in a familiar, easily-installed and easily-manipulated environment.

Our goal in developing CIP (and related projects, an emerging technology we call the *Info-Core Information Infrastructure*) [3, 4] has been to create a generic information infrastructure for integrating scientific and engineering data.

4. CIP Architecture

The CIP system is composed of the following four modules:

- Client Application
- Middleware
- Data Management
- Data Acquisition

The relationships between the CIP modules are represented in Figure 1.

4.1. Client Application

Daily, the MER mission operations staff receives and analyzes the data and status information resulting from

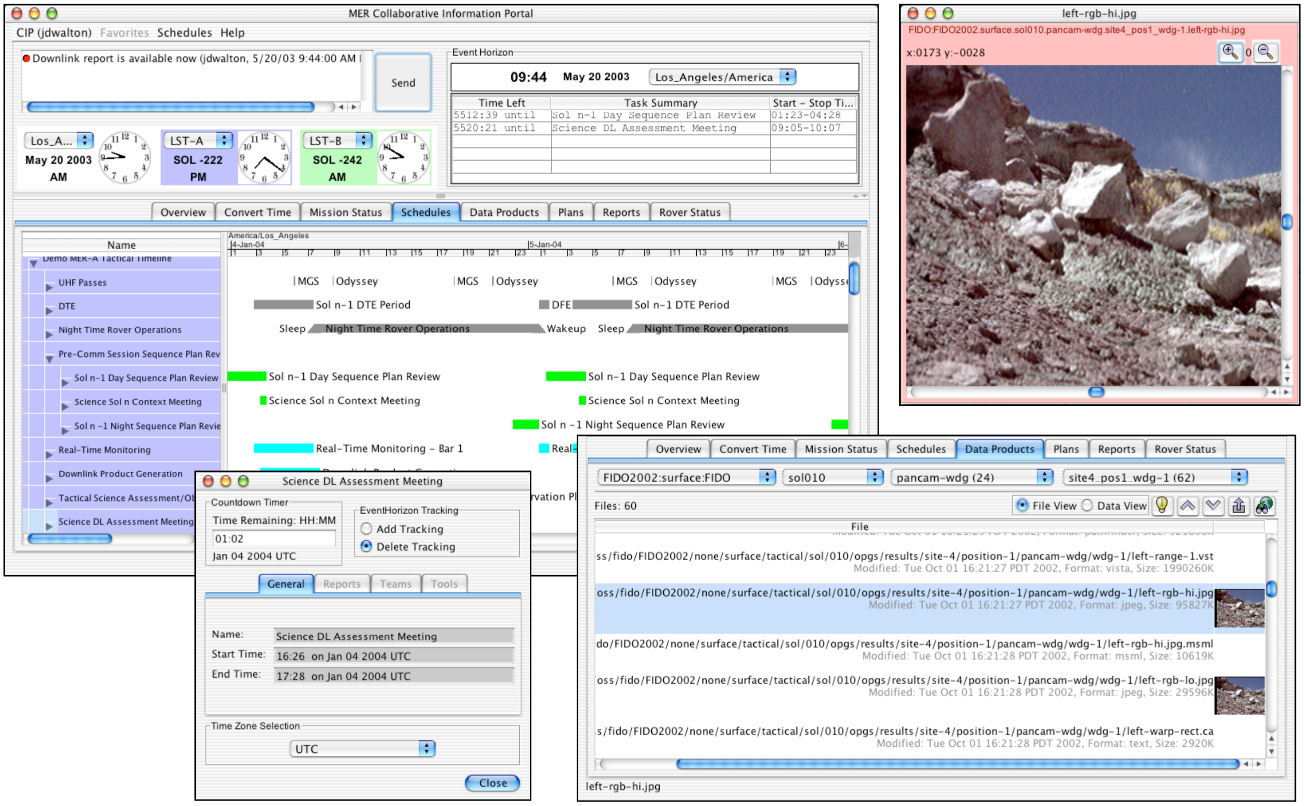

Figure 2. Client application user interfaces.

the rover's activities of the previous day. Using this analysis, scientists create secondary data files and reports and develop the rover plan for the next day.

On a weekly basis, managers and team leads update staffing and operations schedules and develop long-range plans for the rover.

The mission operations staff works in shifts around the clock on Mars time. Critical meetings produce documents that must be handed off to the next shift. The staff is keen to get the latest documents and data files, which are stored on a central file server. Information about those files (meta-data) is coded into the file name.

The goals of the CIP client application are to provide:

- A central place to access mission information.
- A Mars time clock.
- A tool for navigating, searching, and previewing mission data, plans, reports and schedules from various perspectives.
- Notification of new events.
- Automated updates of various mission data files and documents based on subscription.
- Mission broadcast messages.
- Flexibility as mission requirements change.

To meet these goals, the CIP client application consists of several components combined into one application. These include:

- **Clocks**, enabling the user to view any time zone, including the Mars time zones.
- **Science data navigator** for browsing, searching and previewing science data.
- **Reports navigator** for accessing and previewing mission reports.
- **Schedule viewer** for displaying operations and staffing schedules in various time scales including Mars time.
- **Event Horizon** for tracking and counting down to important meetings and other events.
- **Broadcast announcements** for distributing important messages to mission staff.
- **Convert time**, a tool for converting between time zones.

Together these components operate as a cohesive unit, providing situational awareness to the MER mission staff. Figure 2 shows example user interfaces for these components.

The CIP client is a Java application that runs on common scientific user platforms including Sun/Solaris, PC/Windows, and Mac/OS X. This design provides better interactivity than a server-side implementation and makes use of the client host resources. To maintain a

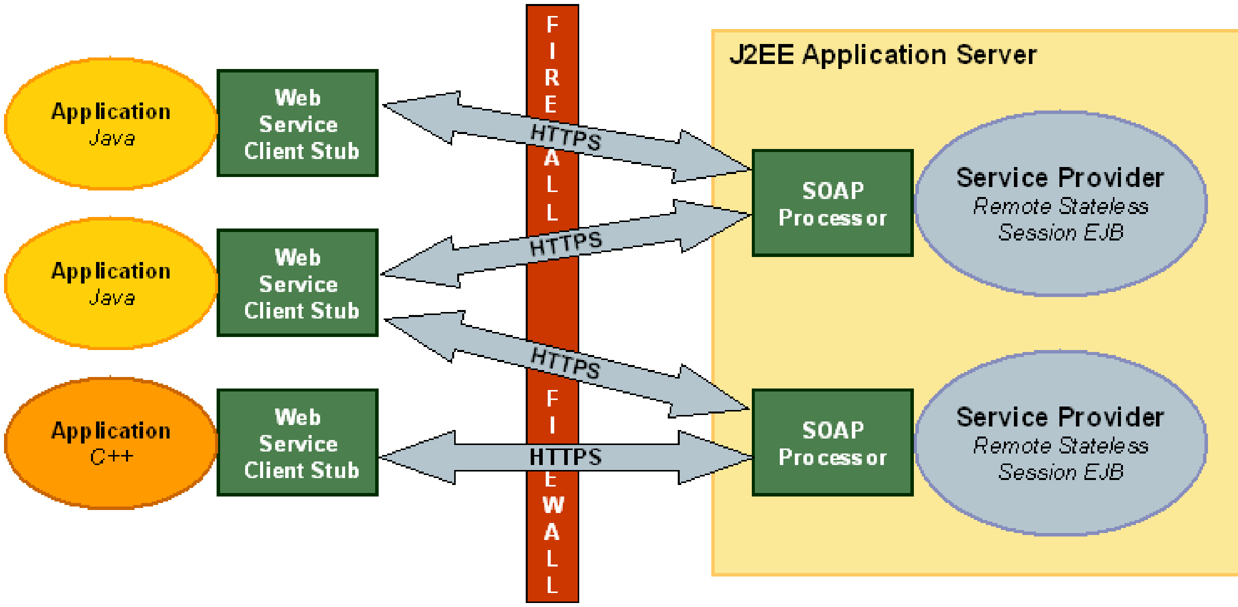

Figure 3. CIP middleware architecture

small resource footprint, it is demand driven and only loads the data the user requests.

4.2. Middleware

No one should notice the middleware when it is doing its job right. Ideally, a user at his or her workstation would believe that the CIP client application accesses data on the server directly and exclusively—he or she should not be aware that, in fact, the middleware is busily fetching data from the backend data stores, caching frequently accessed data, routing asynchronous messages, and serving multiple users simultaneously and securely.

We designed the middleware to meet several key requirements. It was required to:

- Be reliable, scalable, maintainable, and secure.
- Be platform-independent.
- Be built from commercial off-the-shelf (COTS) software.
- Support hundreds of simultaneous users.
- Adhere to industry standards.

Certain mission characteristics also influenced the middleware design:

- Data is downloaded periodically from Mars, with few modifications between downloads. Therefore, the data is mostly read-only.
- Users tend to want to see the new data as soon as it is downloaded, so access frequencies will have large spikes, and the access patterns will have small working sets.
- Compared to e-commerce applications, there are a relatively low number of transactions, but each transaction may involve a relatively large amount of data.
- The mission firewall permits HTTP, but all transactions must be encrypted.
- The CIP client application is written in Java, but some other clients are written in C++.

To satisfy its requirements and to support the mission characteristics, we designed the middleware using industry-standard Java 2 Enterprise Edition (J2EE) technologies [5]. Enterprise JavaBeans (EJBs) provide reliability, scalability, maintainability, and platform-independence [6]. The Java Message Service (JMS) provides asynchronous messaging [7].

We used another industry standard, web services [8], for communications between the middleware and the client applications. Web services are language-independent, allowing us to support both Java and C++ clients. Web services use an XML-based protocol known as SOAP [9]. By transmitting SOAP over HTTPS, we have secure, encrypted communications.

To support the CIP client application and the other clients, the middleware provides a number of services, including:

- **User management services:** User authentication and authorization, session management, and user preferences.
- **Data access services:** Fetch mission data, metadata, and schedules from the backend databases.
- **Time services:** Supply and convert times in various time zones, including current Mars times.

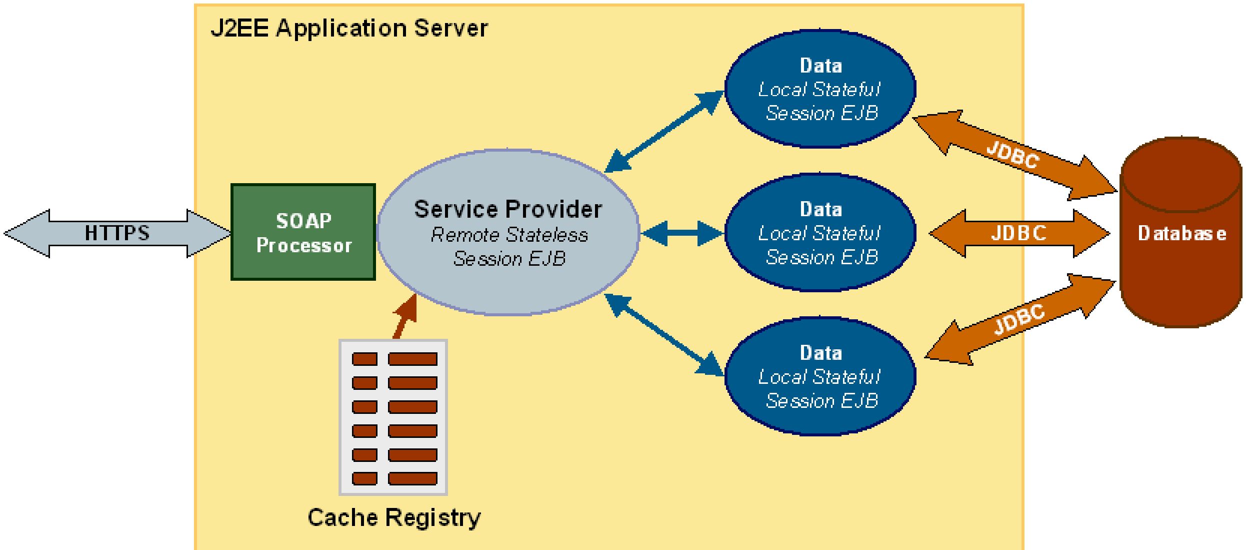

Figure 4. Cached data.

- **File and directory services:** Upload and download mission data and images to and from the backend file stores.
- **Message services:** Asynchronous broadcast announcements and event notification.

Figure 3 shows the middleware architecture.

Each middleware service has a remote stateless session EJB that is the service provider. The application server automatically manages simultaneous calls from multiple clients by maintaining instance pools of these stateless session beans.

To allow the clients to access the middleware via web services, each service provider has a SOAP processor to handle incoming calls. On the client side, each client application has a web services client stub. This stub contains local proxies for the remote APIs of the middleware service providers. Each call to a proxy is automatically converted to a remote call (via SOAP over HTTPS) to the corresponding service provider API.

The data access services use local stateful session EJBs to cache frequently accessed data. This is shown in Figure 4. The application server manages this cache, and least recently used data is automatically removed from the cache whenever the amount of free memory becomes low. The cache registry, a Java hash table, keeps track of what's currently in the cache. Accessing cached data is much faster than fetching it from the remote backend databases via Java Database Connectivity (JDBC) calls [10].

The middleware supports two types of asynchronous

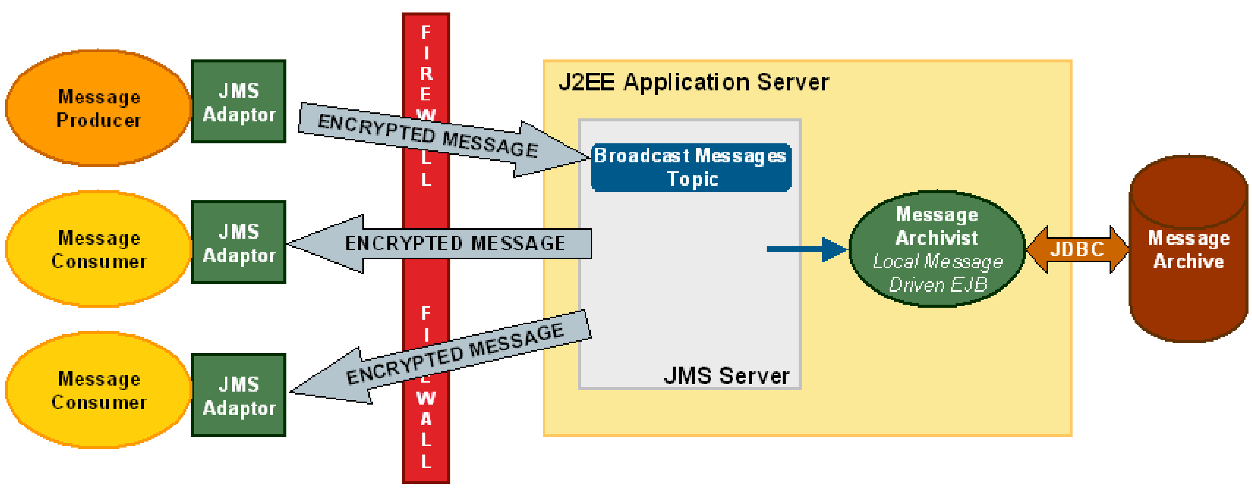

Figure 5. Message broadcasting

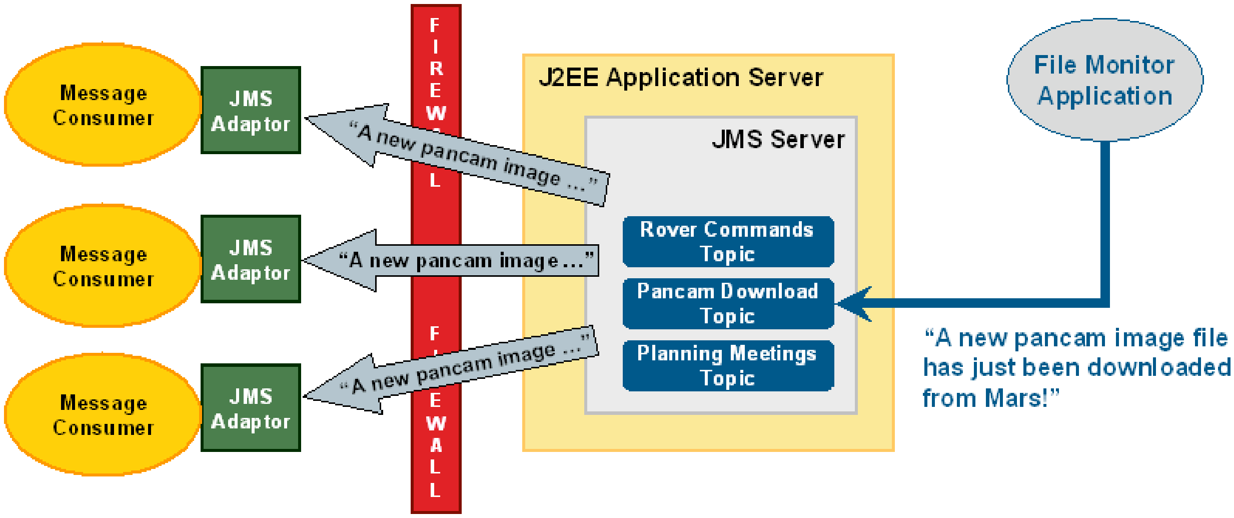

Figure 6. Event notification

messaging within CIP:

- Broadcast announcements that are sent by one user to other users.
- Event notifications that are sent by the middleware to notify clients that certain events have just occurred, such as a download of new data.

For asynchronous messaging, the middleware uses another J2EE technology, the Java Message Service (JMS). JMS topics enable sending messages to multiple clients. Clients subscribe to the topics in which they are interested. A message producer sends a message to a topic, and JMS delivers it to the interested message consumers.

Figure 5 shows message broadcasting. The middleware archives all broadcast messages. The Message Archivist, a message-driven EJB, subscribes to the broadcast messages topic. Therefore, it also receives the broadcast messages, which it then enters into the message archive database.

Figure 6 shows how clients can subscribe to notification topics that correspond to the types of data whose downloads they wish to be notified. A file monitor that runs in the Data Acquisition Module sends a notification message to the appropriate topic. The notification is then forwarded to the interested clients.

The CIP middleware was developed with and is comprised of COTS software. We used Borland's JBuilder 9 Enterprise Edition [11] to develop and debug the Java code, and Apache's open source ANT utility [12] to do the system builds. The Java code adheres to the J2EE standards, including its EJB and JMS interfaces. At runtime, the J2EE application server is BEA WebLogic 8.1 [13]. For security, we installed Verisign certificates [14] into the application server. The WebLogic package includes utilities for generating the web services Java client stubs. To generate the C++ client stubs, we used the open source gSOAP package [15].

4.3. Data Management

The CIP Data Management System is a combination of custom software, a COTS relational database management system (RDBMS) produced by Oracle Corporation (Oracle 9i Enterprise Edition [16]), and a file system. The primary component of the CIP Data Management System is a Product Meta-Database

The Product Met-Database schema captures a hierarchical, relational view of the mission data based on activity along with descriptive information about elements in the hierarchy and pointers to related data products. The functions of the meta-database are:

- Hold sufficient information about the mission data products to support queries on what data products are available, how and when they were created, what system produced them, what their status is and what key parameters are associated with them.
- Perform searches with optimal speed
- Handle new data and updates appropriately

4.4. Data Acquisition

The CIP Data Acquisition module is comprised of Monitor and Loader software. The function of the CIP Monitor is to watch the MER Mission Data Systems and determine when new data have arrived. The function of the CIP Loader is to extract information from specific MER sources and insert it into the Product Meta-Database. The components of the CIP Loader are:

- Parser Modules
- Database Loader

The Parser Modules read specific MER sources and retrieve relevant information for storage in the Product Meta-Database. The Parser Modules format this information into input files for the Database Loader. There may be multiple Parser Modules depending on how many different types of sources from which the CIP needs to pull information. The Parser Modules need to interact with the appropriate MER data systems in order to retrieve the relevant files or information.

The Database Loader takes the input files generated by the Parser Modules and loads their content into the meta-database. The Database Loader is written in Java. It requires the Oracle SQL*Loader to update the Product Meta-Database.

5. Usage Scenarios

As of the writing of this paper, CIP is starting to be used in the MER operational readiness tests (ORTs) in preparation for supporting the actual mission in 2004. CIP has become part of the Mission standard operating procedures for performing some key tasks. When personnel come on shift, they are expected to use CIP to check for announcements, review the daily schedule, and retrieve the latest data products and reports. During the course of the day, the schedule viewer and event horizon in CIP are used on individual workstations and shown on large screen displays for the purpose of alerting personnel to the start and end of key events. As the scientists and engineers collaborate with each other, they use CIP to retrieve key data products and reports and export them to their workstations or to collaborative workspaces, such as the MERBoard [17] where they are discussed, marked up and incorporated into follow-on reports.

6. Future Missions

CIP is based on a general framework for information management systems and as such can be adapted to support other missions beyond MER. The scale of the adaptation effort is dependent on how divergent the future missions goals and data management needs are from those of MER. Often, data repositories vary in structure and format from one mission to another. To adapt to a mission that uses a different style repository from MER, CIP would require a new Data Acquisition module that interfaces with the repository and extracts the key meta-data needed to display and search for data products. In addition, the module might need to employ a different strategy to monitoring the repository for changes. Once the meta-data are stored in the Data Management module, the Middleware and Client Application can operate with the new repository without modifications.

On the client side, we anticipate that future missions will have similar needs for schedule management and data retrieval, although the time management issues will not be as imperative for missions operating on standard Earth time. CIP was developed from the ground up in just under two years, so many of the functions and features that were initially planned had to be set aside in order to meet the challenging schedule. Some examples include advanced data product search capabilities and data mining; automated generation of report summaries; collaborative features such as shared work spaces and information exchange mechanisms; sophisticated notification strategies, including email and pagers; interfaces to complex staff scheduling systems; to name a few. The CIP framework provides a foundation upon which to build all this and more.

7. Acknowledgments

The authors would like to thank the CIP development team for their hard work in making this exciting project a reality and the JPL MER mission management and staff for accepting us onto the mission and facilitating the integration of CIP into the MER ground data systems. In addition we acknowledge NASA's Computing, Information and Communications Technology Program (CICT) program for funding of this work.

8. References

[1] Jet Propulsion Laboratory, *Looking for Signs of Past Water on Mars,* http://mars.jpl.nasa.gov/mer/science/, 2003.

[2] Jet Propulsion Laboratory, *Mission Timeline: Surface Operations,* http://mars.jpl.nasa.gov/mer/mission/tl_surface.html, 2003.

[3] J. D. Walton, R. E. Filman, C. Knight, D. J. Korsmeyer, and D. D. Lee, "D3: A Collaborative Infrastructure for Aerospace Design," *Workshop on Advanced Collaborative Environments,* San Francisco, August 2001.

[4] J. D. Walton, R. E. Filman, and D. J. Korsmeyer, The Evolution of the DARWIN System," *2000 ACM Symposium on Applied Computing*, Como, Italy, March 2000, pp. 971–977.

[5] B. Shannon, *Java™ 2 Platform, Enterprise Edition (J2EE™) Specification, Version 1.3,* Sun Microsystems, August 2001.

[6] L. G. DeMichiel, L. Ü. Yalçinalp, S. Krishnan, *Enterprise JavaBeans™ Specification, Version 2.0,* Sun Microsystems, August 2001.

[7] Sun Microsystems, Inc., *Java Message Service API,* http://java.sun.com/products/jms/, 2003.

[8] Sun Microsystems, Inc., *Web Services,* http://www.sun.com/learnabout/webservices/, 2003.

[9] D. Box, D. Ehnebuske, G. Kakivaya, et al., "Simple Object Access Protocol (SOAP) 1.1," *W3C Note,* May 2000

[10] Sun Microsystems, Inc., *JDBC™ Data Access API,* http://java.sun.com/products/jdbc/, 2003.

[11] Borland, USA, *JBuilder,* http://www.borland.com/jbuilder/, 2003.

[12] Apache Org., *The Apache ANT Project.* http://ant.apache.org/, 2003.

[13] BEA Systems, *BEA WebLogic Server,* http://www.bea.com/framework.jsp?CNT=index.htm&FP=/content/products/server/, 2003.

[14] VeriSign, *VeriSign,* http://www.verisign.com/, 2003.

[15] R. A. van Engelen and K. A. Gallivan, "The gSOAP Toolkit for Web Services and Peer-To-Peer Computing Networks," *Proceedings of IEEE CCGrid Conference,* 2002.

[16] Oracle Corp., *Oracle 9i Database,* http://www.oracle.com/ip/deploy/database/oracle9i/, 2003.

[17] D. Jong, *IBM's BlueBoard Technology on the Red Planet,* http://www.space.com/businesstechnology/technology/merboard_rover_020821.html, August 2002.